SAM-GRID: A System Utilizing Grid Middleware and SAM to Enable Full Function Grid Computing

Andrew Baranovski[1], Gabriele Garzoglio[1], Lee Lueking[1], Dane Skow[1], Igor Terekhov[1], Rodney Walker[2]

1 Fermilab, Batavia, Illinois, U.S.A.
2 Imperial College, London, United Kingdom

Abstract

We present a grid system, which is in development, employing an architecture comprising the primary functional components of job handling, data handling, and monitoring and information services. Each component is built using existing Grid middleware. The Job handling utilizes Condor Match Making Services to broker job submissions, Condor-G to schedule, and GRAM to submit and execute jobs on remote compute resources. The information services provide strategic information of the system including a file replica catalogue, compute availability, and network data-throughput rate predictions, which are made available to the other components. Data handling services are provided by SAM, the data management system built for the Dzero experiment at Fermilab, to optimize data delivery, and cache and replicate data as needed at the processing nodes. The SAM-Grid system is being built to provide experiments in progress at Fermilab the ability to utilize worldwide computing resources to process enormous quantities of data for complex physics analyses.

1 Introduction

Based on the success of SAM [1]-[6], a larger architecture has been conceived that encompasses Grid level job submission and management, and information services. The SAM system was developed at Fermilab to accommodate the high volume data management requirements for Run II physics experiments, and to enable streamlined access and mining of these large data sets. SAM stands for "Sequential Access to data via Metadata" where the "sequential" refers to the layout of physics events stored within files, which are in-turn stored sequentially on tapes within a Mass Storage System. The principle novelties of the larger architecture being developed, when compared with the European Data Grid[7] architecture and other Grid projects include:

- Job Definition and Management: The preliminary job management architecture is aggressively based on the Condor[8] technology provided through our collaboration with the University of Wisconsin CS Group.
- Monitoring and Information Services: We assign a critical role to this part of the system and widen the boundaries of this component to include all services that provide, or receive, information relevant for job and data management.
- Data Handling: The existing SAM Data Handling system, when properly abstracted, plays a principal role in the overall architecture and has direct effects on the Job Management services.

SAM is largely devoted to transparently delivering and managing caches of data. It is the sole data management system in use by the Dzero experiment at Fermilab, and the CDF experiment has identified SAM as a key component of their data handling system, and are currently deploying and beginning to use it. SAM is designed with a distributed architecture, using CORBA as its underlying framework. This has enabled the system to scale to meet the data distribution needs of the Dzero Collaboration which includes 78 institutions and over 600 Physicists located all over the globe. The CDF experiment at Fermilab is currently evaluating SAM, and will probably be heavily involved in the Grid effort at the lab. The DZero and CDF experiments are well underway at Fermilab and their data handling needs are being provided by SAM. Our challenge is the seamless transition of this successful effort to the broader Grid environment, whilst continuing to provide a reliable, production-level service.

2 Goals and Architecture

A major goal of this project is to move toward a more grid-like architecture, using the emergent and standardized Grid components and protocols, without jeopardizing production quality service for the ongoing experimental physics program at Fermilab. It is important to employ the current data handling system because it has been hardened under the stress of heavy use for the last 3 years and includes numerous features required in the HEP environment. The experiments at Fermilab have globally distributed virtual organisations - collaborations of universities and research institutions and it is vital that the needs and constraints of these sites be considered. Since many of these sites support research programs beyond those of Fermilab, for example LHC experiments now under development, it is imperative that this system comply with standards and provide the needed features to access data and perform processing in an opportunistic way, without extensive setups or installations at any site. Cross function with other grids, like EDG, must be supported. Finally, the schedule for this project is short because the needs of the experiments involved are large and imminent.

From the beginning, SAM established a data processing model that enabled users to perform complex chains of processing steps, called applications, on large collections of files referred to as datasets. This coupling of well defined applications with precisely defined datasets is referred to as an analysis project. These concepts were developed to enable users to have straightforward techniques to manage huge tasks, and ensure understandable and reproducible results. In a world of opportunistic computing, monolithic abstractions such as an analysis project need to be decomposed in a manner that will enable grid schedulers to make optimum use of available compute and storage elements.

The general features of the planned architecture are shown in Figure 1, and described in detail in the following sections. The three major components of the architecture are are indicated by the dashed boxes, and the servers are shown as ovals, with arrows showing the information flows throughout the system. The many planned uses of existing, or slightly enhanced, implementations or libraries are indicated by the shaded boxes. Through Dzeros collaboration with PPDG we are able to work closely with the Condor team, and and a major goal is to employ their technology within our system to its fullest extent, as opposed to building our own. We are also using Globus[9] tools, such as MDS, GSI, and GRAM throughout the system. Authentication, Authorization, and Accounting is referred to as AAA, and uses the PKI security implemented in GSI. The FNAL computing division operates a kerberized Certification Authority, to provide user and server credentials.

3 Job Definition and Management

We approach the topic of Job Management from two directions; logical and physical. Logical job management involves understanding what a typical Dzero job is, and how it is viewed by the user and by the scheduler. Physical job management is centred at resource management and involves brokering, scheduling, and planning.

3.1 Job Definition and Structure

In the Grid world of data-intensive computing, it is understood that job specification must minimally contain three parts:

- The input dataset

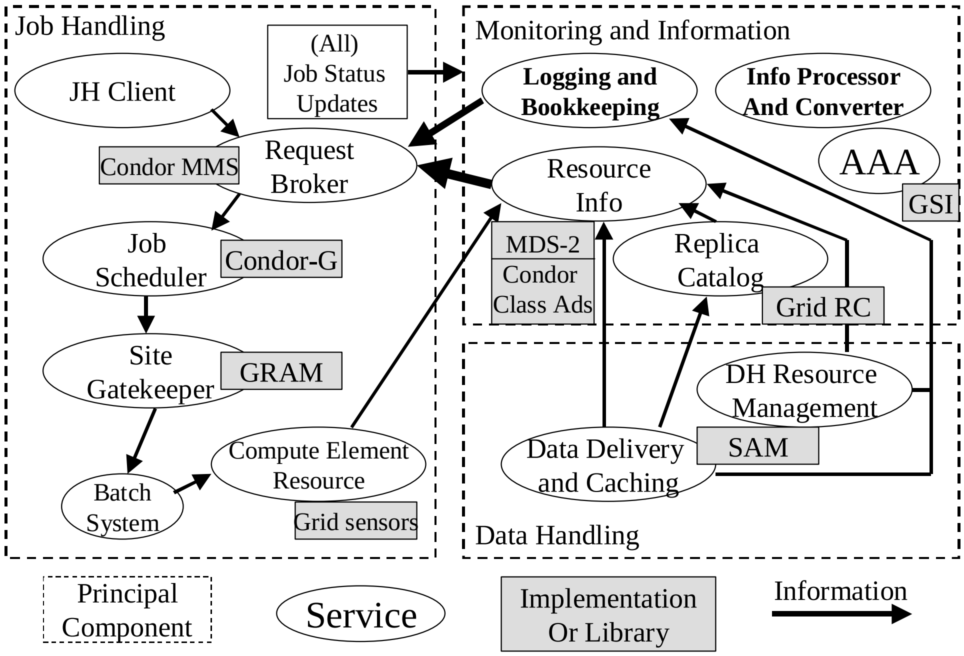

Figure 1. SAMGrid Architecture, including the major components for Job Handling, Data Handling, and Monitoring and Information.

- The application to run. This can be a binary, or a script.
- Some information about the projected output data. For example, the amount, the manner in which it is to be stored or saved, et cetera.

The European Datagrid project uses the terms *input* and *output* sandboxes to represent the incoming and outgoing data. Often, the job is described in Condor's ClassAd [10] language or a derivative thereof. It is further known in the Grid that jobs are composite entities represented by a DAG (Directed Acyclic Graph), and tools such as DAGMan[11] have emerged that can represent dependencies among the jobs parts. In general, however, job definition is more than mere specification of a DAG. The purpose of this section is to reflect the ongoing work to more deeply understand and better describe the Dzero job.

We characterize a job as unstructured if its details are unknown to the scheduler, the status monitoring service, and the metadata catalogue. The user submits a single script to the system even though it may be quite complicated internally and contain distinct steps from the users point of view. The more details of the job that are known to the system, the more structured it is. Work has been to define the job structure of typical Dzero tasks[12]. In summary, a job is a collection of synchronized phases. Within each phase, different lines of execution take place, with varying degrees of parallelism and each consisting of a list of packages to be executed sequentially at each machine.

Implicit for each phase is the input dataset (DS). A phase boundary, as opposed to a boundary between packages in a line, is significant because the user may wish to reuse data created at a later time. Furthermore it is a high-level checkpoint, a place where the job may be suspended and resumed later, perhaps in a different computing location. One should not confuse this check-pointing with the low-level check-pointing offered by Condor. Condor checkpoints a running binary and uses a memory dump of the process, whereas Grid checkpointing means saving results, and re-collecting them prior to job resumption. Since we deal with datasets, inter-phase checkpointing must include the creation of the interim dataset, and likely also storing it with the data handling system. Thus, defining and perhaps storing of the datasets is an integral part of job handling and our final job definition scheme will reflect this concept.

Another key point that cannot yet be fully represented using tools such as DAGMan is the degree of parallelism at each phase or stage. Specifically, an optimal DAG can be specified if the number of nodes is known statically, i.e. prior to job submission. In the SAMGrid project, we believe that once a phase is complete the next phases desired degree of parallelism is determined by the output of the preceding phase. The size of the dataset is one consideration in this regard. Thus, a highly optimized scheduler will obtain this information dynamically with the help of the data handling system. At present, however, we are uncertain whether such a functionality is required in the Grid scheduler or is best left to negotiation between SAM and the local batch system. Both of these, as well as other less important issues, stem from our focus on the efficient data handling available in the SAM system.

3.2 Physical Job Management

The most common Grid infrastructure for the fundamental service of remote job execution and monitoring is provided by the GRAM software from the Globus Toolkit. The Condor-G[13] software provides the higher level service of reliable job submission. For our purposes, we need to provide the request brokering service, referred to as resource brokering elsewhere.

Having reviewed the relevant Grid technologies, we have chosen to use Condors Match Making Service (MMS)[14] , as the request broker. The novelty of our approach is that the properly configured MMS will be the whole request broker, rather than its part, base, or advisor. Our choice is driven by the collaboration of Dzero with the University of Wisconsin under PPDG, as well as by the record of success of MMS as part of the Condor system. Such a utilization of MMS constitutes a paradigm shift for Condor-G from a personal grid manager to a whole system job manager, and for Condor from a local pool manager to a full-fledged Grid manager (note however that the EDG project is independently using Condor components for global brokering as well). This approach gives us unlimited opportunities to implement various policies, as well as resource management considerations together, by means of redefining the ranking functions for the jobs. The exact contents of the ranking function will be determined as a result of our research project. At present, we are confident that the Data Handling (DH) system will play a key role in defining such ranks. We believe the DH will write part of the classAds, and will enter the job management area by means of more than mere publishing of its replica catalogue, which is a more common view in the Grid world. Given the availability of SAM and the experience using it, we believe that we are well positioned to provide a higher degree of sophistication in our computational economy. Examples of factors that may affect scheduling include many things. The utilization, current and projected, of the disk caches managed by SAM stations. The rate of data consumption by the existing projects. The throughput of the data transfers to/from stations (this is different from network conditions as it is affected by the previous item). The near term predictions, from the Data Handling Global Resource manager, on the availability of the remotely cached datasets.

Note that these, and other factors may well change after the job is submitted and before it is successfully matched with resources, thus precluding their full specification in the submitted or the advertised ClassAds. We have therefore requested that the ClassAd mechanism be extended so as to include the capability to call, in the course of the match-making, an externally supplied function. We are using these external functions to consult the data handling system, in order to rank a match according to data location. The Condor MMS is a rather fertile ground for job management. Note that our idea to have the MMS use an external function is not restricted to the data handling considerations. It gives the system designer a wide opportunity to include virtually any arbitrary criteria for scheduling a promising framework for building the prototype for SAM and other Grids.

3.3 Putting it Together for Job Handling

In summary (refer to Figure 1) we envision the following picture. The user defines a job in some user-friendly language, the job definition language(JDL), whereby he specifies the various parts of the job, their dependencies, their desired or possible degree of parallelism and the significance and ways of handling of its interim datasets. This job description file (JDF) is processed by the client (user interface in EDG terminology) so as to incorporate the appropriate ranking functions, a Global Job ID, etc., and translated it into a Condor-JDF (in the future, into DAGMan format). This Condor-JDF is then translated by the submission layer into a ClassAd, which waits in a queue periodically polled by the request broker. The MMS of the request broker matches the job with the available resources and evaluates the ranks of the matches. The job ClassAd is modified to include the contact information of the highest ranking resource and then it is sent to the submission service for dispatch. It is then executed in the well-known fashion by the lower level services.

4 Information and Monitoring

There are two main aspects of the management of information and monitoring, one for publishing and discovering of resources, and one for human monitoring of the system and of its individual jobs. These are intimately related but allow initial separation of concerns. In the prototypical stage of our project, we plan for both to use the Globus Monitoring and Discovery Service (MDS), which is based on the Grid Monitoring Architecture[15].

In this respect, our project is quite similar to all other Grid projects and we do not give much attention to it here. We hope to converge on a common way to define the systems resources, using a standard schema such as the Glue schema[16].

One important comment, which may be relevant only to the prototype phase of the project, concerns the way we gather and publish information from the producers. Since we use Condors MMS to broker the requests, we will use its ClassAds mechanism as a perfectly viable way to represent resources in the system, an option not widely recognised in the wide Grid community. On the other hand, since we plan to use the MDS technology for the front-end (user-monitoring), well need to publish such information into MDS. Consequently, we envision information converters between the MDS language and the ClassAd language, a task that is hopefully commonly solved in the future.

5 Data Handling with SAM

SAM has been built to maximize data and processing throughput for the systems where it is deployed. Adapters have been built for several batch systems (LSF,PBS, FBS and Condor), and a fair-share scheduling and resource co-allocation algorithm is used to maximize throughput and bring data and processing resources together temporally. It is desirable to exploit these features in the larger architecture, although it is not yet understood at which levels in the overall system this is best done. A part of SAM that has not been fully developed is its site optimization server, or DH Resource Management. This component will provide the service of coordinating access to data stored at other SAM stations or in Mass Storage Systems. Data access will be optimized based on the policy that will ensure optimal use of the fabric resources available, such as network, tape drives, and tape mounts.

In order to work within this architecture, SAM will need to be augmented with the capabilities to use the standard Grid protocols to publish and access needed information. SAM exists as a set of cache management and data transfer services that are distributed among participating sites. Each functional set of these services is referred to as a SAM Station. It currently maintains its configuration and replica catalogue for all stations in a central database. This will be expanded to include the information services discussed above, which will maintain replica information local to each site. SAM may be required to produce Condor Class Ads that will be used by the Request Brokering service. SAM will likely receive information from MDS relating to the job scheduling changes and this will be used as input to modify data transfer and caching decisions.

6 Status and Plans

To begin the effort we have established several projects to understand the Globus and Condor software. We are enabling the Globus Security Infrastructure in our client server system. GridFTP was introduced as a transport protocol for SAM and tested using the DOE CA for data transfers to and from sites in the UK including Imperial College and Lancaster. A test bed was established to experiment with job submission using the Globus toolkit and Condor-G. Understanding the capabilities of these tools is fundamental to make the appropriate decisions when designing and establishing the architecture for remote job dispatch and SAM. The test bed has three machines at Fermilab; one at Imperial College and, one at the University of Texas, Arlington. Job submission was tested from Condor-G/GRAM to PBS, and Condor. UTA has provided an LSF test system and performed successful tests. We are actively working with the Condor and Globus teams and providing feedback on problems and compatibility issues we discover.

We are working with the Condor Team, in order to use the Negotiator, which is part of Condor, as the Match Making Service for the Job Brokering Service. We have agreed on a set of new features that are needed for our project (some mentioned in Section 3.2), and it is now being tested. The immediate needs of this project, the Condor features required to implement a Request Broker as a Condor Match-Making Service, have been delivered, and we look forward to further cooperation in the future. The next step will be enabling the experiments with the scheduling of data-intensive jobs on the Dzero Condor-G testbed, and we now have a prototype system. Further, we will seek to tune the resources and job requirements specifications as to achieve high throughput on the system as well as implement the policies of the experiments on resource allocation and usage. We hope to study the emerging issues and design solutions. Interaction of the job scheduler with a mature data handling system in the context of a live experiment will likely become an exciting hands-on research project.

At the same time we are deploying an MDS-based information system. It will be used for monitoring the status of the fabric and the services running on the grid, as well as more dynamic information such as the status of a job. Aside from being useful for the system administrators who tune their resource usage, this global information will be the source of input to the Condor MMS, which will in turn match the users' descriptions of jobs with the most suitable resources available at the time.

In addition, effort is continuing within Dzero to understand the structure of a generic job (Section 3.1), such that a formal Job Definition will result, based on the physicists use cases and experience running SAM jobs. We will build a Job Description Language, and in doing so we will desire a new thread of the collaborative effort with the Condor team. Furthermore, as D0 approaches understanding of complex jobs and develops a need to schedule and execute them, a meta-scheduling infrastructure will be required similar to, but likely richer than, the existing DAGMan in Condor. We are likely to propose an effort to enhance DAGMan accordingly, as this will be our initial tool of choice for complex jobs scheduling.

7 Acknowledgements

We would like to thank the Fermilab Computing Division for its ongoing support of SAM, especially the ODS, D0CA, and ISD Departments. We would like to thank everyone at D0 who has contributed to this project, and the many important discussions we have had there. We would like acknowledge Vicky White for her past contributions into the project, continuing support of the project today and for the many fruitful discussions with her that have shaped our ideas for the future. We thank Ruth Pordes for her tireless support for this project and her quality management in PPDG. Imperial College London and The University of Texas Arlington have provided hardware and manpower resources enabling the successful operation of the distributed testbed, and their contributions are important parts of this effort. We also thank Rick St Denis and other CDF collaborators for the continuing interest and feedback to SAM.

This work is sponsored by DOE contract No. DE-AC02-76CH03000. Additional funding for Grid related SAM work is provided by the DOE SciDAC program through Dzero's association with the Particle Physics Data Grid (PPDG) collaboration. The Imperial College effort is funded through GridPP in the UK.

References

[1] The SAM team, A. Baranovski, D. Bonham, C. Jozwiak, L. Loebel Carpenter, L. Lueking, C. Moore, H. Schellman, I. Terekhov, J. Trumbo, S. Veseli, M. Vranicar, S. White, http://d0db.fnal.gov/sam

[2] Terekhov et. al., SAM for Dzero A Fully Distributed Data Access Layer for Dzero Run II, The VII International Workshop on Advanced Analysis Techniques in Physics Research (ACAT 2000), October 2000.

[3] Loebel Carpenter et. al., SAM Overview and Operational Experience at the Dzero Experiment, Resource Management in SAM and the D0 Particle Phys, CHEP 2001, September 2001, Beijing China.

[4] Igor Terekhov, Distributed Processing and Analysis of Physics Data in the Dzero SAM System at Fermilab, Fermilab-TM-2156.

[5] I. Terekhov et al., Distributed Data Access and Resource Management in the Dzero SAM System, High Performance Distributed Computing Conference, August 6-10, 2001.

[6] A. Baranovski, et. al., SAM Managed Cache and Processing for Clusters in a Worldwide Grid-Enabled System, FERMILAB-TN-2175, May 2002.

[7] The European Datagrid project, http://www.eu-datagrid.org

[8] The Condor project home page, http://www.cs.wisc.edu/condor/.

[9] The Globus Project, http://www.globus.org .

[10] Classified Advertisements, http://www.cs.wisc.edu/condor/classad/

[11] DAG Manager, http://www.cs.wisc.edu/condor/dagman/

[12] D. Meyer et al., D0 Job Components, privte communication, unpublished.

[13] J. Frey, T. Tannenbaum, M. Livny, I. Foster, S. Tuecke, Condor-G: A Computation Management Agent for Multi-institutional Grids, in proceedings of The 10-th IEEE International Symposium on High Performance Distributed Computing, August 2001, San Francisco, CA.

[14] R. Raman, M. Livny and M. Solomon, Matchmaking: Distributed Resource Management for High Throughput Computing, in Proceedings of the Seventh IEEE International Symposium on High Performance Distributed Computing, July 28-31, 1998, Chicago, IL.

[15] K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman. Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10), IEEE Press, August 2001.

[16] The Glue schema (work in progress)
http://www.hicb.org/glue/glue-schema/schema.htm